Radio Shack®

MICROCOMPUTER NEWSLETTER

700 One Tandy Center
Fort Worth, Texas 76102

OCT 1979

PRICES MAY VARY AT INDIVIDUAL STORES AND DEALERS

Fort Worth Scene

Format Changes in the Microcomputer Newsletter

During the next two months you will see a few format changes in the Microcomputer Newsletter. We hope to institute several regular features, including FORT WORTH SCENE and TRS-80 CLASSROOM. We will also give you a short index for each issue, and publish an annual cumulative index. The first of these cumulative indexes will be published in the December issue, and will cover all newsletters published through December 1979. TRS-80 CLASSROOM will begin several series of articles designed specifically for users of Level I, Level II, TRSDOS, assembly language, and Model II. As soon as FORTRAN is released we will begin articles on FORTRAN as well. One other change which we will make is to begin a Volume #, Issue # format to help you keep track of Newsletters. Our thanks to Brian Jaffee of Trans-World Agency, Inc. for this suggestion.

It is our intention to publish one Newsletter every month, and to make them something you will save for reference. We are removing direct advertising from the Newsletter itself. We will continue to bring you information and reviews on new hardware and software, but this will be done in a separate pull-out section.

If you have any suggestions on content or form for the Microcomputer Newsletter, please direct them to us at:

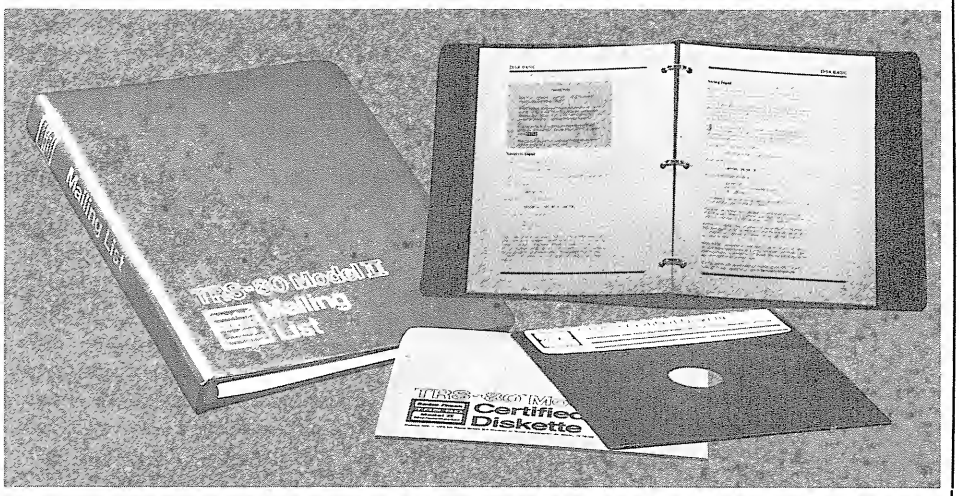
MICROCOMPUTER NEWSLETTER
700 ONE TANDY CENTER
FORT WORTH, TEXAS 76102

Notes on Previous Newsletters

AUG/SEPT 1979—In the article on screen graphics to lineprinter from Mr. Thorpe, we should have commented that the program prints 128 columns. If your printer does not print that many columns, you will lose part of the screen.

JULY 1979 — Concerning the two programs to compact and speed cassette I/O on Level-II, one change you can make is to eliminate lines 40 and 130, then make the following changes:

(Continued on page 4)



TRS-80™ MODEL II SOFTWARE – Each package includes a deluxe binder with diskette holder, diskette(s) and complete instructions.

Model II Software Availability Update

The following dates are the current projections for the availability of TRS-80 Model II software:

26-4501	General Ledger	End Oct.
26-4502	Inventory Control	End Dec.
26-4503	Payroll	January
26-4504	Accts. Receivable	January
26-4506	Mailing List	Mid-Nov.

Integrating TRSDOS™ 2.2 Information into TRSDOS Manual

In order to integrate the TRSDOS 2.2 release information into our current TRSDOS manual, renumber the release pages as follows:

Cover Letter-1.7

Memo	New #	Memo	New #
1	1-8	2	3-2.5
3	1-9	4	1-10
5	4-10.5	6	7-2.5
7	7-2.7	8	7-2.81
9	7-2.82	10	7-2.83
11	7-10.5	12	7-18.51
13	7-18.52	14	5-10
15	5-11	16	5-12
17	5-13	18	5-14
19	5-15	20	5-16
21	1-11		

Computer Services Address and Phone Numbers

Computer Services
900 Two Tandy Center
Ft. Worth, Texas 76102

Computer Services
Phone Numbers:

1-800-433-1679 (WATS except Texas)
1-800-772-5914 (WATS inside Texas)
1-817-390-3583 (Switchboard)

All TRS-80 related calls and mail should be directed to the above address, or one of the above phone numbers. Computer Services is equipped with knowledgeable people who are there to answer your questions. If they do not have an immediate answer, they have the internal contacts to get the answers in a minimum amount of time. Questions sent to the Newsletter must be sent to Computer Services via internal mail, which simply delays your response.

Newsletter Index

VOLUME #1 ISSUE #8

TRS-80 CLASSROOM

TRS-80 users have expressed confusion over the cause and cure of "garbage digits" which appear in double precision numbers. The purpose of this article is to precisely describe the problem and suggest some remedies.

Try typing the following into a Level II or a Disk TRS-80:

```
>A=.1
>B=.5
>PRINT A,B
  .1                    .5
```

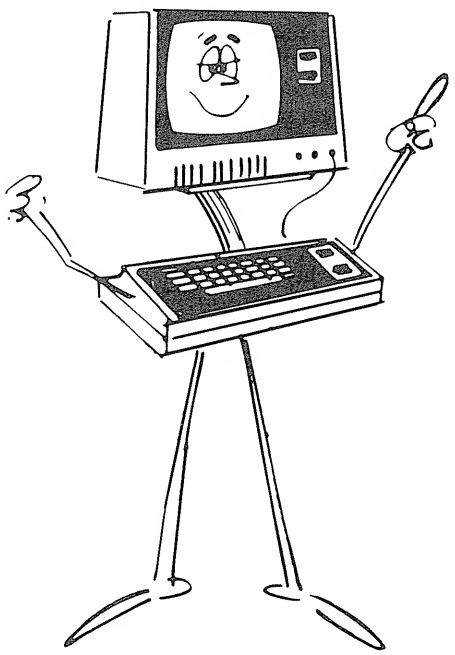
This is, of course, what you would expect. Now enter .1 and .5 as double precision values:

```
>A#=.1
>B#=.5
>PRINT A#,B#
  .1000000014901161     .5
```

.1 (Which can not be represented exactly in binary) did not convert to a double precision value correctly.

Here are three methods to make sure that BASIC knows you want a double precision value. Each of the following three lines is a different method.

```
>A#=0.1D
>B#=0.100000000
>A=.1:C#=VAL(STR$(A))
>PRINT A#,B#,C#
   .1                .1                .1
```



The first two methods are for numbers you actually type in (as part of the BASIC program). You must put a 'D' on the end of the number, or type more than seven digits. This will make BASIC use a double precision value.

The third method is the one most people miss. Here you force BASIC to take a single precision variable and re-interpret the value as double precision. Integer values (whole numbers) are no problem in double precision arithmetic. Look at the following program:

```
10 HOURS=12
20 OVERTIME=0
30 IF HOURS>8 THEN OVERTIME=
   HOURS-8:HOURS=HOURS-
   OVERTIME
40 RATE=5.65
50 R#= VAL(STR$(RATE))
60 H#=VAL(STR$(HOURS))
70 O#=VAL(STR$(OVERTIME))
80 PAY#= H# * R# + O# * 2 * R#
90 PRINT PAY#
```

On lines 10 through 40, only single precision values (and whole numbers) are used, so no problems will occur. But in line 80, we want double precision PAY# out of single precision HOURS, OVERTIME, and RATE. This is where problems arise. In lines 50 to 70, we fix all the single precision numbers into double precision values. Because of lines 50 to 70, lines 80 and 90 will produce the correct gross pay. Note that the '2' is used on line 80 with no ill effect. Remember, numbers without fractional parts (nothing after the decimal point) will not bother double precision arithmetic. If the overtime rate were 1.5 instead of 2, A '1.5D' should be used to insure that the correct double precision value is used.

Disk BASIC users can use a 'DEF FN' statement to make programming easier. The following program repeats the one above, but uses a 'DEF FN' function which fixes double precision numbers:

```
5 DEF FNX#(A)=VAL(STR$(A))
10 HOURS=12
20 OVERTIME=0
30 IF HOURS >8 THEN
   OVERTIME=
   HOURS-8:HOURS=HOURS-
   OVERTIME
40 RATE= 5.65
50 PAY#= FNX#(HOURS)*
   FNX#(RATE)+FNX#(OVERTIME)
   *2*FNX#(RATE)
60 PRINT PAY#
```

Simply put the FNX#( ) around any variable or constant that has a decimal portion. If you are experienced in binary arithmetic, you should be able to tell when you need to apply this fix function. When in doubt, put FNX# around any number you question.

Why do "garbage digits" appear? The answer lies in the way BASIC stores numbers and converts them from single precision to double precision.

Recall our first two numbers, A and B. The internal hexadecimal representations of these two numbers are:

Decimal		Hexadecimal			
.1	=	125	76	204	205
.5	=	128	0	0	0

Note the zeros on the end of the .5 representation in hexadecimal. This means that .5 is not an infinite repeating fraction in hexadecimal (or binary). But, when .1 is converted to hexadecimal, it becomes an infinite repeating fraction. In binary it is 0.11001100110011001100 . . . Now let's look at the hexadecimal values for A# and B#.

Decimal		Hexadecimal							
.1	=	125	76	204	205	0	0	0	0
.5	=	128	0	0	0	0	0	0	0
					(4 zeros added to end)				

When BASIC converts a single precision number (.1 or .5) to a double precision value, you'll note that four zeros were simply added to the end of the single precision representation. Since .5 was not an infinite repeating binary fraction (It has zeros on the end), adding more zeros to the end does not affect the value. The number .1, however, is no longer correct. The correct hexadecimal value is:

125 76 204 204 204 204 204 205.

Why didn't BASIC figure out that the last four numbers were not zeros? The answer is , you can't get something out of nothing. BASIC must be explicitly told what the last four numbers are, or it will use zeros (because BASIC simply doesn't know).

Budget Management (26-1603) Note

Budget Management requires a 110 column (or wider) printer, such as our Line Printer III (26-1156) or Line Printer I (26-1152).

User Programs and Hints

Mr. Gerard M. Foley of Canal Winchester, Ohio sent us these program lines to be used with our program to merge two cassette programs (July 1979). These two subroutines, and appropriate modifications from you to fit your own data, will allow you to enter data into a program, add program lines, modify the program and not lose the data you have entered! Almost anyone who has developed a program knows how frustrating it can be to enter all of the data into a program only to find that a close quote mark was left out of a PRINT statement. As soon as you put the close quote in, BASIC erases all of your data and you have to start reentering.

The following are quotes from Mr. Foley's letter:

"Before entering your program, reserve space by typing a number after "MEMORY SIZE?". If you type 30000 (in a 16K Level II machine) you will reserve space for more than 600 single precision numbers. Then enter the following program, in addition to the program you are developing (see "How to Merge Two Programs" Page 2, July 1979 Newsletter):

```
60000 I=0:A=J
60010 GOSUB 60200
60020 FOR I= 1 TO J
60040 A=W(I)
60050 GOSUB 60200
60060 NEXT I
60070 FOR I=J+1 TO 2*J
60080 A=X(I-J)
60090 GOSUB 60200
60100 NEXT I
60110 RETURN

60200 JK=4*I+30000
60210 FOR I1= 0 TO 3
60220 POKE(JK+I1),
      PEEK(VARPTR(A)+I1)
60230 NEXT I1
60240 RETURN

61000 I=0
61010 GOSUB 61200
61020 J=A
61030 FOR I=1 TO J
61040 GOSUB 61200
61050 W(I)=A
61060 NEXT I
61070 FOR I=J+1 TO 2*J
61080 GOSUB 61200
61090 X(I-J)= A
61100 NEXT I
61110 RETURN

61200 A=0
61210 JK=4*I+30000
61220 FOR I1=0 TO 3
61230 POKE(VARPTR(A)+I1),
      PEEK(JK+I1)
61240 NEXT I1
61250 RETURN
```

"J items of data are stored in each of the two arrays W(I) and X(I). When subroutine 60000 is called, the number J and the arrays are stored in protected memory from 30000 on up. After program modification, if subroutine 61000 is called, the data will be restored to J and the two arrays."

"If you wish to store and recover only one array, delete 60070-60100 and 61070-61100. If you wish to store more than two arrays, add additional 'FOR' loops like 60070-60100 and 61070-61100."

"In using the programs, as soon as valid data have been entered in the main program, whether from the keyboard or from a cassette, call subroutine 60000 above to save it. Whenever a valid alteration to the data has been made, call 60000 again. Then if you wish to remove a bug, call 61000 after the change. Remember that 'CLEAR' statements will erase the data from your main program (but not from the protected storage), so put the GOSUB 61000 after such statements."

"If you decide to reserve more or less storage by entering a number other than 30000 after "MEMORY SIZE?", put the same number in statements 60200 and 61210 above. Multidimensional arrays can be saved and recovered by appropriate modifications to the 'FOR' loops 60030-60100 and 61030-61100."

Mr. Foley has set up his routines to handle single precision numeric arrays. If you study the 'VARPTR' statement, you will see that only minor modifications are needed to handle integer and double precision numerics, and that the same type routine will also allow you to save alphanumeric data. This routine should certainly save some of us a lot of time. Lines 5-200 provide you with a short program to dump 400 single precision numbers and recover them.

```
 5 CLEAR:DIM W(200),X(200)
10 FOR I=1 TO 200
 15 PRINTI
 20 W(I)=I
 30 X(I)=I
 40 NEXT
 45 J=200
 50 GOSUB 60000
 60 STOP
 65 CLEAR
 70 DIM W(200),X(200)
 80 FORI=1 TO 200
 90 PRINTW(I);X(I),
100 NEXT
110 GOSUB 61000
120 FOR I=1 TO 200
130 PRINTW(I);X(I),
140 NEXT
200 END
```

Model II User's Note

Using a Serial Printer With TRSDOS

Output to a serial printer can easily be accomplished with TRSDOS 1.1 and the following routine called "PRINTER/BAS." Note: 'LPRINT' and 'LLIST' will not function with a serial printer. These features will be implemented in a later version of TRSDOS.

To Use "PRINTER/BAS":

1. Under TRSDOS, initialize Channel A with "SETCOM," using the parameters which fit your printer.

2. "LOAD COMSUB" under TRSDOS.

3. When you call "BASIC" reserve memory for "COMSUB" by using: BASIC -M:61000

"PRINTER/BAS" should be included, as a subroutine, in any BASIC program which will output to a serial printer. Before using "GOSUB 10000" you must set two variables. "WW" is an integer. If "WW" is any positive number, the string "WW$" will be output to the printer. If "WW" is zero, the program will output a carriage return and line feed (if you do not need the line feed, DELETE 10210). If "WW" is a negative value, the program will output "WW" spaces to the printer. In effect, a negative "WW" serves as a TAB function. The value "WW$" contains the information to be printed (up to 255 characters).

"PRINTER/BAS"

```
10000  REM
10010  REM     ENTRY INTO THIS
10015  REM       ROUTINE RE-
10020  REM       QUIRES THAT WW$
10025  REM       CONTAIN THE
10030  REM       CHARACTER OR
                 STRING TO BE
                 PRINTED.

10040  REM     THE VARIABLE WW
10045  REM       IS USED TO DE-
10050  REM       TERMINE THE
                 FOLLOWING:

10060  REM     IF WW IS NEGATIVE
10070  REM       —OUTPUT
10080  REM       ABS(WW) SPACES

10090  REM     IF WW IS ZERO—
10100  REM       OUTPUT CAR-
10105  REM       RIAGE RETURN
10110  REM       LINE FEED

10120  REM     IF WW IS POSITIVE —
                 OUTPUT WW$
10130  REM       STRING
10140  REM
10150  REM
10160  REM
```

(Continued on page 4)

Radio Shack
COMPUTER MERCHANDISING
700 ONE TANDY CENTER
FORT WORTH, TEXAS 76102

IF UNDELIVERABLE DO NOT RETURN

FORT WORTH SCENE

(Continued from page 1)

```
60 IF  LEN(B$+STR(D(I))+A$)
   >240 THEN PRINT#-1, N,B$:
   B$=""
80 NEXT: IF B$<>"" THEN
   PRINT#-1, N,B$
140 INPUT#-1, NN, B$:
   N=LEN(B$):LF=1
```

These changes will give you one less record to write, and increase the number of items stored in each record.

One other thing, we found that we needed to 'CLEAR 1000' in order to make the program run.

Model II User's Note

(Continued from page 3)

```
10170  DEFUSR0 = &HEF80
10180  IF SGN (WW)=-1 THEN 10230
10190  IF SGN(WW)<>0 THEN 10270
10200  RX$=USR0(CHR$(13))
10210  RX$=USR0(CHR$(10)):REM
       IF NO LINE FEED NEEDED,
       DELETE THIS LINE
10220  RETURN
10230  FOR RQ=1 TO ABS(WW)
10240  RX$=USR0(CHR$(32)) :REM
       BLANKS
10250  NEXT RQ
10260  RETURN
10270  FOR RQ=1 TO LEN(WW$)
10280  RP$=MID$(WW$,RQ,1)
10290  RX$=USR0(RP$)
10300  NEXT RQ
10310  RETURN
```

Model I to Model II Transfer Software Now Available

Software (ACT-0131 $20.00) is now available from National Parts which enables you to transfer software from TRS-80 Model I to TRS-80 Model II. This software allows you to transfer BASIC programs which have been saved in ASCII, data files, and machine language programs. The software set includes a 5½" disk for Model I and an 8" disk for Model II. The minimum Model I configuration is a one disk 32K system with RS-232C.

Please note that while you may transfer Model I BASIC programs to Model II, they will need some modification before they will run properly. The main items which require attention are:

1. 'IF-THEN' Statements — Model II requires the use of 'THEN' where it was optional on Model I.
2. 'PRINT@' Statements — Most of your 'PRINT@' statements will need to be changed to properly format screens on the larger Model II video.
3. Drive Specifications — At least for a while, most Model II users will be running single disk systems. This requires that you change any drive specs which refer to drives other than drive 0 (zero).
4. All references to 'PEEK', 'POKE', 'INP', or 'OUT' must be removed. These commands are not needed in Model II BASIC. Instead, TRSDOS supervisor calls are available which will allow you to perform the same functions.

Other changes can be made which will enhance your programs and take better advantage of Model II's capabilities, but the four mentioned above are required if your software is to run properly.

Transfer of programs between Model I and Model II requires a special cable and terminator plug. These items are described in your Model II manual.

The terminator plug is described on page 1 of the Corrections to Model II Owner's Manual dated 8/22/79 which is included with all Model II Manuals. The cable is described on page 144 of the Model II TRSDOS Manual and updated on page 1 of the Corrections.

These items will be available from National Parts (AW-2440 $71.00) in approximately 45 days.

If you do not wish to purchase these items due to the relatively small amount of transfering that you need to do, contact your nearest Radio Shack Computer Department or Computer Center. These facilities are being set-up to help you make the Model I to Model II transfers. The personnel in these locations will be familiar with the software, and with proper transfer techniques, and will be happy to get you started and provide any assistance you may need.

Your local Radio Shack store will be able to order the transfer materials for you, or provide you with the location of the nearest Computer Department or Computer Center.

Inventory Control Users Please Note:

Inventory Control System I, (26-1553) requires a 132-column printer. If you are about to purchase a line printer to go with ICS, do *not* buy our Line Printer II (26-1154). Instead, buy our Line Printer III (26-1156) or Line Printer I (26-1152).

The printouts in the ICS manual appear to be 80 columns wide, because they were printed on a Line Printer I (26-1152) with character width narrowed.